IN THE CLAIMS

Please cancel claims 8 and 9.

Please amend the claims as follows.

1. (Currently amended) An apparatus for computer hardware multithreading comprising:

a plurality of processors, each processor having hardware support for the capability of executing a plurality of threads;

a memory coupled to the plurality of processors; and

a thread dispatch mechanism residing in the memory and executed by at least one of the plurality of processors, the thread dispatch mechanism determining which of the plurality of processors are idle, which of the plurality of processors is busy processing a thread but can accept ~~an additional~~ a new thread, and which of the plurality of processors cannot accept ~~an additional~~ the new thread since it is working on a maximum number of threads the processor can execute and, the thread dispatch mechanism dispatching [[a]] the new thread to an idle processor, if one exists.

2. (Currently amended) The apparatus of claim 1 wherein, if none of the plurality of processors is idle and if at least one of the plurality of processors can accept ~~an additional~~ the new thread, the thread dispatch mechanism dispatches the new thread to one of the plurality of processors that can accept ~~an additional~~ the new thread.

3. (Currently amended) The apparatus of claim 1 wherein, if all of the plurality of processors cannot accept ~~an additional~~ the new thread, the thread dispatch mechanism waits for one of the plurality of processors to complete processing a thread, thereby becoming a processor that can accept ~~an additional~~ the new thread, and then dispatches the thread to the processor that can accept ~~an additional~~ the new thread.

4. (Currently amended) A method for dispatching threads in a computer system that includes a plurality of processors that can each support hardware multithreading to execute a plurality of threads, the method comprising the steps of:

(1) determining the status of each of the plurality of processors, wherein a processor is idle if not executing any threads, wherein the processor can accept ~~an additional~~ a new thread if busy working on one or more threads but has the capacity to process the ~~additional~~ new thread, and wherein the processor cannot accept ~~an additional~~ the new thread if busy working on a maximum number of threads the processor can execute; and

(2) dispatching [[a]] the new thread to an idle processor, if one exists.

5. (Currently amended) The method of claim 4 further comprising the step of:

if none of the plurality of processors is idle and if at least one of the plurality of processors can accept ~~an additional~~ the new thread, the thread dispatch mechanism dispatches the new thread to one of the plurality of processors that can accept ~~an additional~~ the new thread.

6. (Currently amended) The method of claim 4 further comprising the steps of:

if all of the plurality of processors cannot accept ~~an additional~~ the new thread, the thread dispatch mechanism waits for one of the plurality of processors to complete processing a thread, thereby becoming a processor that can accept ~~an additional~~ the new thread, and then dispatches the thread to the processor that can accept ~~an additional~~ the new thread.

7. (Currently Amended) A computer-readable program product comprising:

(A) a thread dispatch mechanism that determines which of a plurality of processors in a hardware multithreading, multiprocessor computer system are idle, which of the plurality of processors is busy but can accept ~~an additional~~ a new thread, and which of the plurality of processors cannot accept ~~an additional~~ the new thread since it is working on a maximum number of threads the processor can execute, the thread dispatch mechanism dispatching [[a]] the new thread to an idle processor, if one exists, wherein each processor can execute a plurality of threads; and

(B) ~~computer readable signal bearing~~ recordable media bearing the thread dispatch mechanism.

8. (Cancelled)

9. (Cancelled)

10. (Currently Amended) The program product of claim 7 wherein, if none of the plurality of processors is idle and if at least one of the plurality of processors can accept ~~an additional~~ the new thread, the thread dispatch mechanism dispatches the new thread to one of the plurality of processors that can accept ~~an additional~~ the new thread.

11. (Currently Amended) The program product of claim 7 wherein, if all of the plurality of processors cannot accept ~~an additional~~ the new thread, the thread dispatch mechanism waits for one of the plurality of processors to complete processing a thread, thereby becoming a processor that can accept ~~an additional~~ the new thread, and then dispatches the new thread to the processor that can accept ~~an additional~~ the new thread.

12. (Previously presented) The apparatus of claim 1 wherein all processors are made busy with a first thread before dispatching a second thread to any processor.

13. (Previously presented) The method of claim 4 wherein all processors are made busy with a first thread before dispatching a second thread to any processor.

14. (Previously presented) The program product of claim 7 wherein all processors are made busy with a first thread before dispatching a second thread to any processor.